*Script for latest ChatGPT models and equivalent. Compulsory user input is in purple.*

# Preparation

* Work through each task list in the given order during the entire conversation.
* You are {INJECT\_CHATBOT}, a language model developed by {INJECT\_CHATBOT\_COMPANY}. Imitate a MEDIUM TEMPERATURE setting of 0.9 (for a creative yet structured approach, encouraging new ideas without losing coherence) and a VERY HIGH TOP\_P setting of 0.9 (promoting diversity in the responses while ensuring logical connections within the generated content). Provide ALL RESPONSE CONTENT without asking questions during the entire conversation. DO NOT print any keys (example: Use "example" AND NOT "{key-example}") during the entire conversation. Make sure to replace the keys inside "{}" with their values. Use a "scientific tone" during the entire conversation, unless instructed otherwise. Do not confuse roles and stakeholders, they are different.
* Memorise "Unified Modelling Language" as {key-uml}. Memorise "Agent-Based Social Simulation (ABSS) Study" as {key-studyType}.

# Analysis

### Problem Statement

* Take on the "role" of a "{INJECT\_DOMAIN\_RELATED\_ROLE}" with experience in "Agent-Based Social Simulation" as {key-role1}. Memorise this {key-role1}.
* Define the "topic" of the memorised {key-studyType} as "{INJECT\_TOPIC}". Memorise this topic as {key-topic}.
* Using a "scientific and inspirational tone", define a novel and creative "context" for the memorised {key-topic} in 200 WORDS (if possible). Memorise this context as {key-context}.
* Memorise "{INJECT\_RESEARCHDESIGN}" as {key-researchDesign}. Memorise "{INJECT\_DOMAIN}" as {key-domain}. Memorise "{INJECT\_SPECIALISATION}" as {key-specialisation}.
* Define 5 "stakeholders" for the memorised {key-topic}, to participate in a co-creation roleplay game. Memorise these stakeholders together with their personas as {key-stakeholders} (you do not need to create names for personas). Explain how your answers differ for different memorised {key-stakeholders}.
* Write a Markdown document (separate each section using headers). Display "Problem Statement" as markdown 'Heading Level 3 ###'. Display: 1) Memorised {key-role1}. 2) Memorised {key-topic}. 3) Memorised {key-researchDesign}. 4) Memorised {key-domain}. 5) Memorised {key-specialisation}. 6) Memorised {key-context}. 7) Memorised {key-stakeholders}.
* Play a co-creation role-play game in which all the memorised {key-stakeholders} discuss with each other potential aims for the study considering the pros and cons. Keep a "debating tone" but be creative with the format, order, structure and style of the discussion/debate. The moderator focuses on 1 novel RANDOM question. List 2 potential aims that satisfy the viewpoints of all participating memorised {key-stakeholders}. Memorise these potential aims as {key-potentialAims}. Propose 3 criteria for ranking the 2 potential aims to support the decision which aim to carry forward. Define 5 "keywords" for the memorised {key-studyType} in the context of the memorised {key-topic} in the form of a comma-separated list. Memorise these 5 keywords as {key-keywords}.
* Using an inspirational tone, define a brief "title" for the memorised {key-studyType} in the context of the memorised {key-topic} in 12 WORDS (if possible). Memorise this title as {key-title}.
* Define the "aim" for the memorised {key-studyType} in the context of the memorised {key-topic} in 40 WORDS (if possible). Use the memorised {key-potentialAims} in your definition. Memorise this aim as {key-aim}.
* Using exactly the same markdown code as what was generated before, append the following sections/chapters and display the full markdown code: 1) Memorised {key-title}. 2) Memorised {key-aim}. 3) Memorised {key-keywords}.

### Study Outline

* Now, write a new, different Markdown document (separate each section using headers). First, output "Study Outline" as markdown 'Heading Level 3 ###'. Then list: 1) A definition of the term "objective" in the context of the memorised {key-studyType} in 1 concise sentence. 2) A definition of the term "hypothesis" in the context of the memorised {key-studyType} in 1 concise sentence. 3) A definition of the term "experimental factor" in the context of the memorised {key-studyType} in 1 concise sentence. 4) A definition of the term "output" in the context of the memorised {key-studyType} in 1 concise sentence.
* Simulate and play a co-creation role-play game in which all the memorised {key-stakeholders} discuss with each other potential Agent-Based Social Simulation (ABSS) objectives for the study considering the pros and cons. Keep a "debating tone" but be creative with the format, order, structure and style of the discussion/debate. The moderator focuses on 1 novel RANDOM question. Provide the question and the details of the controversial discussion. Agree on a few potential ABSS objectives that satisfy the view of all participating memorised {key-stakeholders}. Memorise these potential ABSS objectives as {key-potentialObjectives}.
* Propose 3 criteria for ranking the potential ABSS objectives to support the decision which objectives to carry forward.
* Define 2 "ABSS objectives" for the memorised {key-studyType} in the context of the memorised {key-topic}. Use the memorised {key-potentialObjectives} in your definitions. List the objectives with 2 relevant performance measures for each objective. Memorise these 2 objectives together with the performance measures as {key-objectives}.
* Play a new co-creation role-play game in which all the memorised {key-stakeholders} discuss with each other potential ABSS hypotheses for the study considering the pros and cons. Keep a "debating tone" but be creative with the format, order, structure and style of the discussion/debate. The moderator focuses on 1 novel RANDOM question. Provide the question and the details of the controversial discussion. Agree on 4 potential ABSS hypotheses that satisfy the view of all participating memorised {key-stakeholders}. Memorise these potential ABSS hypotheses as {key-potentialHypotheses}. Propose 3 criteria for ranking the 4 potential ABSS hypotheses to support the decision which hypotheses to carry forward.
* Define 2 "ABSS hypotheses" and 2 relevant performance measures for the memorised {key-studyType} in the context of the memorised {key-topic}. The hypotheses MUST not be related to the memorised {key-objectives}. Use the memorised {key-potentialHypotheses} in your definitions. Memorise these 2 hypotheses AND the performance measures as {key-hypotheses}.
* Play a new co-creation role-play game in which all the memorised {key-stakeholders} discuss with each other potential ABSS experimental factors for the study considering the pros and cons. The ABSS experimental factors must be relevant to the memorised {key-hypotheses} and memorised {key-objectives}. Keep a "debating tone" but be creative with the format, order, structure and style of the discussion/debate. The moderator focuses on 1 novel RANDOM question. Provide the question and the details of the controversial discussion. Agree on 6 potential ABSS experimental factors that satisfy the view of all participating memorised {key-stakeholders}. Memorise these potential ABSS experimental factors as {key-potentialExperimentalFactors}. Then propose 3 criteria for ranking the 6 potential ABSS experimental factors to support the decision which experimental factors to carry forward.
* Define 3 "ABSS experimental factors" for the memorised {key-studyType} in the context of the memorised {key-topic}. You ALWAYS must satisfy the following 2 requirements for defining experimental factors: 1) The experimental factors need to be useful for creating memorised {key-studyType} scenarios. 2) CONSIDER the memorised {key-objectives} and the memorised {key-hypotheses} for defining the experimental factors. MAKE SURE TO use the memorised {key-potentialExperimentalFactors} in your definitions. List the experimental factors with 1 value range for each experimental factor. 1 of them MUST use a 'nominal scale' AND 1 of them MUST use an 'ordinal scale' AND 1 of them MUST use a 'ratio scale'. Memorise these 3 experimental factors together with the value ranges as {key-experimentalFactors}.
* Play a new co-creation role-play game in which all the memorised {key-stakeholders} discuss with each other potential ABSS outputs for the study considering the pros and cons. Keep a "debating tone" but be creative with the format, order, structure and style of the discussion/debate. The moderator focuses on 1 novel RANDOM question. Provide the question and the details of the controversial discussion. Agree on 6 potential ABSS outputs that satisfy the view of all participating memorised {key-stakeholders}. Memorise these potential ABSS outputs as {key-potentialOutputs}. Propose 3 criteria for ranking the 6 potential ABSS outputs to support the decision which outputs to carry forward.
* Define 3 "ABSS outputs" for the memorised {key-studyType} in the context of the memorised {key-topic}. You ALWAYS must satisfy the following 2 requirements for defining outputs: 1) Some outputs need to be useful for measuring if the memorised {key-objectives} have been satisfied. 2) Some outputs need to be useful for accepting or rejecting the memorised {key-hypotheses}. Use the memorised {key-potentialOutputs} in your definitions. List the outputs and explain links to the memorised {key-objectives} OR the memorised {key-hypotheses} in 1 concise sentence each. Memorise these 3 outputs together with the links as {key-outputs}.
* Now, write a new, different Markdown document (separate each section using headers). First, output "Study Outline" as markdown 'Heading Level 3 ###'. Then list: 1) Memorised {key-potentialObjectives} and the criteria for ranking them. 2) Memorised {key-objectives}. 3) Memorised {key-potentialHypotheses} and the criteria for ranking them. 4) Memorised {key-hypotheses}. 5) Memorised {key-potentialExperimentalFactors} and the criteria for ranking them. 6) Memorised {key-experimentalFactors}. 7) Memorised {key-potentialOutputs} and the criteria for ranking them. 8) Memorised {key-outputs}.

### Model Scope

* Now take on the "role" of a "Senior Software Developer" with experience in "Unified Modelling Language (UML)", memorise this role as {key-role2}. DO NOT CONFUSE {key-role2} and {key-stakeholders}, they are different concepts.
* Now, write a new, different Markdown document (separate each section using headers). First, output "Model Scope" as markdown 'Heading Level 3 ###'. Then list: 1) The memorised {key-role2}. 2) A definition of the term "model scope" in the context of the memorised {key-studyType} in 1 concise sentence. 3) A definition of the term "UML actor" in the context of the memorised {key-uml} in 1 concise sentence.
* Play a co-creation role-play game in which all the memorised {key-stakeholders} discuss with each other potential some ABSS UML actors for the study considering the pros and cons. Keep a "debating tone" but be creative with the format, order, structure and style of the discussion/debate. The moderator focuses on 1 novel RANDOM question. Provide the question and the details of the controversial discussion. Agree on 8 potential ABSS UML actors that satisfy the view of all participating memorised {key-stakeholders}. Memorise these potential ABSS UML actors as {key-potentialUMLActors}. Propose 3 criteria for ranking the 8 potential ABSS UML actors to support the decision which ABSS UML actors to carry forward.
* Define 4 ABSS UML actors as USERS OF THE SYSTEM described in the memorised {key-topic}. Use the memorised {key-potentialUMLActors} in your definitions. Memorise these 4 UML actors together with a persona description as {key-umlActors}.
* Create a Markdown table for the following: Define 15 "real-world elements" with relevance to the memorised {key-topic}. You ALWAYS must satisfy the following 8 requirements for defining real-world elements: 1) Consider what 'real-world elements' are needed to represent in the model scope and to satisfy the memorised {key-aim}. 2) ALL 4 memorised {key-umlActors} MUST BE REPRESENTED. 3) At least 2 Physical Environment elements MUST be present. At least 2 Social Aspect elements MUST be present. At least 2 Psychological Aspect elements MUST be present. At least 2 Miscellaneous elements MUST be present. 4) Consider the memorised {key-context}. 5) Consider all nouns in the conversation history. 6) Each element can only be in 1 category. 7) Social Aspect elements MUST describe theories of social behaviour. 8) Psychological Aspect elements MUST describe theories of psychological behaviour. Feel free to be creative and add your ideas. Categorise the 'real world elements' into Actors, Physical Environment, Social Aspects, Psychological Aspects, and Miscellaneous. TABLE MUST include 15 rows. Organise all 15 elements into categories and provide a brief explanation. Memorise these 15 elements and explanations as {key-explanations}.
* List the memorised {key-topic} relevant real-world elements in the form of table rows. Provide a column for Category. Provide a column for Sub-Category. Provide a column with the memorised {key-explanations}. Provide a column with concise justifications in ABOUT 25 WORDS. Memorise this table as {key-modelScope}.
* Create a Markdown table for the following: Define 4 models for implementing elements of the memorised {key-modelScope}. Provide 1 social model AND 1 behavioural model AND 1 psychological model AND 1 technical model. Find relevant theoretical models in the SCIENTIFIC LITERATURE. Provide a full EXISTING UP-TO-DATE scientific paper (conference or journal) or book REFERENCE in HARVARD STYLE for each in a separate column. Memorise these 4 model details together with a description and the relevant reference as {key-implementationModels}.
* Now, write a new, different Markdown document (separate each section using headers). First, output "Model Scope" as Markdown 'Heading Level 3 ###'. Then list: 1) Memorised {key-role2} 2) Definitions of: "model scope" in the context of the memorised {key-studyType} and "UML actor" in the context of the memorised {key-uml}. 3) Each of the memorised {key-umlActors}, with their personas 4) The memorised {key-modelScope} table 5) The memorised {key-ImplementationModels} table.

### Key Activities

* Now, write a new, different Markdown document (separate each section using headers). First, output "Key Activities" as markdown 'Heading Level 3 ###'. Then list: 1) A definition of the term "user story" in the context of the memorised {key-uml} in 1 concise sentence. 2) A definition of the term "use case" in the context of the memorised {key-uml} in 1 concise sentence.
* Create a Markdown table for the following: Define 2 "UML user stories" for each of the 4 memorised {key-umlActors} (example: As an 'actor' I want 'action' so that 'achievement'). Memorise ALL 8 UML user stories as {key-umlUserStories}. Translate the memorised {key-umlUserStories} into UML use cases. Memorise ALL 8 UML use cases as {key-umlUseCases}. List ALL 8 memorised {key-umlUserStories} and ALL 8 corresponding memorised {key-umlUseCases} side by side in two columns inside the table sorted by memorised {key-umlActors}. Memorise this table as {key-umlUseCaseTable}.
* Generate a script for a "comprehensive use case diagram" in "Mermaid.js". Use the memorised {key-umlActors} as UML actors. Use the memorised {key-umlUseCases} as UML use cases. You ALWAYS must satisfy the following 4 requirements for defining the use case diagram: 1) Each UML actor MUST be linked to at least 1 UML use case. 2) Each UML use case MUST be linked to at least 1 UML actor OR MUST be pointing to at least 1 other UML use case. 3) There is no UML actor to UML actor interaction. 4) A UML use case CAN be linked to multiple UML actors. Add relationships with 'detailed descriptors'. Start the script with `graph LR`. DO NOT Add subgraphs. Use the following formats; for actor: `Actor1((actor\_name))`, for use case: `Activity1([activity\_name])`, for relationship: `Actor1 -->|activity\_description| Activity1`. Feel free to be creative and add your ideas. Memorise this Mermaid.js script as {key-mermaidKeyActivitiesScriptDraft}.
* Build upon the {key-mermaidKeyActivitiesScriptDraft}. Add ADDITIONAL use cases directly linked to the UML actors and ADDITIONAL use cases LINKED to other USE CASES. Link use cases for different actors. IMPROVE clarity of descriptors. Critically REFLECT and IMPROVE the script based on your reflection. Find and remove any Mermaid.js script errors. Memorise this "Mermaid.js" script as {key-mermaidKeyActivitiesScript}.
* Now, write a new, different Markdown document (separate each section using headers). First, output "Key Activities" as Markdown 'Heading Level 3 ###'. Then: 1) Display the memorised {key-umlUseCaseTable} 2) Display the memorised {key-mermaidKeyActivitiesScript}.

# Design

### Archetypes

* Now, take on the additional third role of an experienced "Management Expert". The memorised {key-stakeholders} remains the same. Memorise this role as {key-role3}. DO NOT CONFUSE {key-role3} and {key-stakeholders}, they are different concepts.
* Now, write a new, different Markdown document (separate each section using headers). First, output "Archetypes" as Markdown 'Heading Level 3 ###'. Then: 1) Display memorised {key-role3} 2) List: definitions of "archetype" and "categorisation schema" in the context of the memorised {key-role3} in 1 sentence.
* Play a co-creation role-play game in which all the memorised {key-stakeholders} discuss with each other potential archetypes for each of the memorised {key-umlActors} individually. Keep a "debating tone" but be creative with the format, order, structure and style of the discussion/debate. Provide details of the discussion and provide 6 potential archetypes FOR EACH of the 4 memorised {key-umlActors}. Then include 3 criteria to identify them. Agree on 2 potential archetypes FOR EACH of the memorised {key-umlActors} that satisfy the view of all participating memorised {key-stakeholders}. Memorise these potential archetypes as {key-potentialArchetypes}.
* Create a Markdown table for the following: Define 4 categorisation schemata, 1 for each of the 4 memorised {key-umlActors}. You ALWAYS must satisfy the following 5 requirements for defining categorisation schemata: 1) Each of the 4 tables must be based on memorised {key-umlActors} behaviour, preferences, characteristics, demographics, habits, and the likelihood of actions. 2) Each of the 4 tables MUST contain 3 characteristic rows. 3) Characteristics inside a table MUST use 1 'nominal scale' AND MUST use 1 'ordinal scale' AND MUST use 1 'ratio scale'. 4) Characteristics inside a table MUST provide value ranges for these scales. 5) Table columns: Actor Category, Individual Characteristic, Scale, Value Range. CONSIDER the memorised {key-potentialArchetypes} in your definitions. Memorise ALL 4 categorisation schemata as {key-categorisationSchemata}.

### Agent & Object Templates

* Take on the additional "role" of a "Senior Software Developer" with experience in the "Unified Modelling Language". The memorised {key-stakeholders} remains the same. Memorise this role as {key-role4}. DO NOT CONFUSE {key-role4} and {key-stakeholders}, they are different concepts.
* Now, write a new, different Markdown document (separate each section using headers). First, output "Agent and Object Templates" as Markdown 'Heading Level 3 ###'. Then list: 1) Memorised {key-role4} 2) Definitions of "class", "class diagram", "state chart", "state variable" in the context of the memorised {key-uml}.
* Generate a script for a 'comprehensive class diagram' in "Mermaid.js". Start the script with `classDiagram`. \*\*Define a class for each of the Actor and Physical Environment categories in the memorised {key-modelScope}\*\*. Add collective classes for individual actors where appropriate (example: bird > flock). DO NOT create abstract classes. Delete all getter and setter methods. The Main class must be called `ArtificialLab`. The `ArtificialLab` class must have array attributes for EACH Actor object and EACH Physical Environment object. The `ArtificialLab` class must contain appropriately named attribute names and methods to measure statistics for the memorised {key-objectives} AND verify the memorised {key-hypotheses}. Generate the attributes, methods and relationships necessary to perform all {key-umlUseCases}. Add inheritance, composition, aggregation and association relationships. Feel free to be creative and add your ideas. Memorise this Mermaid.js script as {key-mermaidClassDiagramScriptDraft}.
* Build upon the {key-mermaidClassDiagramScriptDraft}. Account for all the {key-umlUseCases} by adding additional attributes, methods and missing relationships (association, aggregation, composition, inheritance) between classes. Critically REFLECT and IMPROVE the script based on your reflection. Make sure to include all memorised {key-experimentalFactors} and memorised {key-outputs} in the appropriate classes. Find and remove any Mermaid.js script errors. Memorise this Mermaid.js script as {key-mermaidClassDiagramScript}.
* \*\*For EACH INDIVIDUAL of the 4 memorised {key-umlActors}, generate separate 'comprehensive state machine diagrams' in "Mermaid.js"\*\*. Use `stateDiagram-v2`. Define their states and state transitions between these states. Add text to the transitions to describe what they represent (example: `s1 --> s2: Generate A transition`). You ALWAYS must satisfy the following 2 requirements for each state machine diagram: 1) ALL diagrams MUST have AT LEAST 1 entry transition (example: `[\*] --> s1`) AND 1 exit transition (example: `s1 --> [\*]`). 2) Provide a memorised {key-uml} note for every individual state (example: `note left of [actual state] : Informative text note`). Memorise this Mermaid.js script as {key-mermaidStateMachineDiagramsScriptDraft}.
* Build upon the state machines you generated. Add additional states and additional TRANSITIONS. Add compound states. Provide a memorised {key-uml} NOTE for every individual state, explaining the related state (example: `note left of [actual state] : Informative text`). Critically REFLECT and IMPROVE the script based on your reflection. Memorise this Mermaid.js script as {key-mermaidStateMachineDiagramsScript}.
* Create a Markdown table for the following: Iterate through the memorised {key-mermaidStateMachineDiagramsScript} and define up to 3 variables FOR EACH diagram for keeping track of continuous changes of agent and object states (often a level of something: example 'tiredness level'). Create a "state variables table" with all state variables (columns: state machine diagram, variable, unit, definition of variable). An example row in the table: "State machine shopper, satisfaction level, scale 1-10, represents the satisfaction level". Memorise this state variables table as {key-stateVariablesTable}.
* Create a Markdown table for the following: Create a "state transitions table" with ALL STATE TRANSITIONS FROM EVERY STATE DIAGRAM (columns of the table are: actor, start state, end state, type of transition, detail). Detail MUST be 1 concise sentence. Possible TYPE OF TRANSACTION: timeout, condition, rate. Memorise this state transitions table as {key-stateTransitionsTable}.

### Interactions

* Now, write a new, different Markdown document (separate each section using headers). First, output "Interactions" as Markdown 'Heading Level 3 ###'. Then: 1) Display definition of the term "sequence diagram" in the context of {key-uml} in 1 concise sentence.
* Generate a script for a 'comprehensive sequence diagram' in "Mermaid.js". Use the memorised {key-mermaidClassDiagramScript} to get all relevant ACTORS and OBJECTS. \*\*Define interactions between the 'different actors' and 'actors and objects' FOR ALL memorised {key-umlUseCases}\*\*. Use cases should be presented as NOTES on the vertical axis above each use case representation, in this format: `note over <TheActorName>: <meaningful\_note\_here>`. Actors and objects should be as lifelines on the horizontal axis. Use `actor` (NOT `participant`) for ALL ACTORS (example: `actor ActorName`). DO NOT use aliases. Present ACTIVATIONS and DEACTIVATIONS for actors and objects on the LIFELINES. Each use case should be connected to the corresponding sequence of events. Add the prefix "The" to all ACTOR and OBJECT names. IGNORE the "ArtificialLab". An example interaction with a label is as followers `TheActor1->>TheActor2: <add\_informative\_label\_here>`. Memorise this Mermaid.js script as {key-mermaidSequenceDiagramScriptDraft}.
* Build upon the {key-mermaidSequenceDiagramScriptDraft}. ADD loops, alternative paths, optional paths and parallel interactions. Critically REFLECT and IMPROVE the script based on your reflection. ENSURE that ALL memorised {key-umlUseCases} have been considered. Memorise this Mermaid.js script as {key-mermaidSequenceDiagramScript}.

# Implementation

* Now, you are a GAML (GAMA Markup Language) software developer. You will generate a GAML script for the GAMA simulation platform in a step-by-step prcoess according to my prompts. Use the contextual knowledge you have gathered. Firstly, create a `global` block which contains the necessary GAML code to initialise global variables and implement global methods from the ArtificialLab as reflexes. The `global` block should also instantiate (create) the `species` (`species` is used to define the actors/systems/species from {key-mermaidClassDiagramScript}). The `global` block is created as follows: ```gaml\nmodel <gaml\_model\_name>\n\nglobal {\n\t// <declaration and initialisation of global attributes e.g. float x <- 0.5;>\n\t// global reflexes ...\n\treflex foo {\n\t\t// <logic for each global method from ArtificialLab>\n\t\t...\n\t}\n\tinit {\n\t\t// create species\n\t\tcreate <species\_name\_1> number: <initial\_population\_of\_species\_name\_1>;\n\t\t...\n\t}\n}\n...\n```. Output the GAML script so far and save as {key-gamlScriptDraft1}.
* Build upon {key-gamlScriptDraft1}. Use `species` to define the actors/systems/species from {key-mermaidClassDiagramScript}. DO NOT implement the ArtificialLab. At this stage, each `species` must include ALL required attributes. Make sure ALL species necessary to perform the intended simulation have been implemented. DO NOT implement any methods/actions/reflexes inside the species at this stage. An example species scaffold can be found below: ```gaml\nmodel <gaml\_model\_name>\n\nglobal {\n\t// <declaration and initialisation of global attributes e.g. float x <- 0.5;>\n\t// global reflexes ...\n\treflex foo {\n\t\t// <logic for each global method from ArtificialLab>\n\t\t...\n\t}\n\tinit {\n\t\t// create species\n\t\tcreate <species\_name\_1> number: <initial\_population\_of\_species\_name\_1>;\n\t\t...\n\t}\n}\nspecies <species\_name> {\n\t// <declaration and initialisation of attributes e.g. float probability <- 0.5;>\n}\n```. Output your full GAML script so far and save as {key-gamlScriptDraft2}.
* Now, give the pseudocode and all steps to implement the logic for \*\*ALL\*\* methods inside the classes in {key-mermaidClassDiagramScript}, the methods will be implemented later as a reflex or action in the appropriate species. Think about the necessary interactions between classes/species for each method and include them in your pseudocode.
* Build upon {key-gamlScriptDraft2}. Implement the steps/pseudocode for all the methods inside the correct species; use `action` if the behaviour that is not expected to be invoked/executed at each timestep, or use `reflex` if the method is expected to be executed at each timestep. \*\*Make sure every `species`, `reflex` and `action` in the memorised {key-gamlScriptDraft2} is fully implemented, YOU MUST THINK CAREFULLY AND STEP BY STEP WHEN IMPLEMENTING THE LOGIC\*\*. \*\*Make sure every method declared in classes in {key-mermaidClassDiagramScript} is implemented as a `reflex` or `action` in the appropriate species\*\*. \*\*If any of the reflexes/actions need to interact with another species, use interactions in {key-mermaidSequenceDiagramScript} and relationships in {key-mermaidClassDiagramScript} to connect the respective species using attributes, reflexes and actions\*\*. \*\*Implement interactions between agents, for example, add the following in an appropriate `action` or `reflex` block inside `speciesA` to interact with `speciesB`: ```gaml\nspecies A {\n\taction <species\_A\_action\_name> {\n\t\task speciesB {\n\t\t\tdo <species\_B\_action\_name>;\n\t\t}\n\t}\n}\n```\*\*. \*\*The body of each `action` and `reflex` must be implemented FULLY\*\*. If the action should return a value, replace `action` keyword with return type. Output your full GAML script so far and save as {key-gamlScriptDraft3}.
* Build upon {key-gamlScriptDraft3}. Generate an `experiment {...}` block, with `output {...}` and `display {...}` blocks inside it. The code must provide the necessary attributes and logic to allow users to manipulate all {key-experimentalFactors}, make sure to include memorised {key-experimentalFactors} as `parameter` variables in the `experiment` block, so its values can be modified, each parameter must have a corresponding variable in the `global` block. \*\*You must implement all the code necessary for the simulation to meet the memorised {key-objectives} and experiments must allow for the user to verify the memorised {key-hypotheses}\*\*. Make sure to include performance metrics for the memorised {key-objectives} and {key-hypotheses}. Each `species` (and the `global` block) must include all required attributes and logic to compute all {key-outputs}. The `experiment` block must display all {key-outputs}. Any species you want to visually display must be declared in the `display` block in `experiment` and must have a corresponding `aspect` in their species block, in the format: `aspect <aspect\_name e.g. base> {\n\tdraw <shape e.g. circle>(size) color: #color;\n}`. If {key-outputs} would benefit from supplementary txt/csv files, also add logic to output them. A GAML script scaffold for reference: ```gaml\nmodel <gaml\_model\_name>\n\nglobal {\n\t// <declaration and initialisation of global attributes and global variables corresponding to parameters e.g. float x <- 0.5;>\n\t// global reflexes ...\n\treflex foo {\n\t\t// <logic for each global method from ArtificialLab>\n\t\t...\n\t}\n\tinit {\n\t\t// create species\n\t\tcreate <species\_name\_1> number: <initial\_population\_of\_species\_name\_1>;\n\t\t...\n\t}\n}\n\nspecies <species\_name\_1> {\n\t// <declaration and initialisation of attributes e.g. float probability <- 0.5;>\n\treflex <reflex\_name> {\n\t\t// <logic>\n\t}\n\taction <action\_name>(<parameters\_if\_any>) {\n\t\t// <logic>\n\t}\n\taspect <aspect\_name e.g. base> {\n\t\tdraw <shape e.g. circle>(size) color: #color;\n\t}\n}\n\nspecies <species\_name\_2> {\n\t...\n}\n\n...\n\nexperiment <experiment\_name> {\n\t// e.g. to slow the simulation... float minimum\_cycle\_duration <- 0.7;\n\t// e.g. set seed if randomness is used... float seed <- 1.0;\n\tparameter <parameter\_name> var:<name\_of\_corresponding\_global\_variable> <parameter\_options e.g. min:, max:, category:>;\n\toutput {\n\t\tdisplay <display\_1\_name> {\n\t\t\t// if you want to visually show species\n\t\t\tgrid gridName (if grid to display)\n\t\t\tspecies <species\_to\_visually\_display> aspect: <aspect\_name>;\n\t\t}\n\t\t// display other outputs separately (e.g. charts)...\n\t\tdisplay <display\_2\_name> refresh:every(5#cycles) {\n\t\t\tchart "Chart" type: histogram/pie/series/scatter/xy/radar/heatmap/box\_whisker {\n\t\t\t\tdata "Number of Z" value: <output\_attribute\_number\_of\_z>;\n\t\t\t\t...\n\t\t\t}\n\t\t}\n\t}\n}\n```. Output your full GAML script so far and save as {key-gamlScriptDraft4}.
* Build upon {key-gamlScriptDraft4}. Carefully scan through the GAML code, find missing parts of code and any mistakes (such as stubbed actions/reflexes, unimplemented logic inside comments/todos, missing interactions between species, incomplete and unimplemented global methods (reflexes) from the ArtificialLab etc.), resolve them by implementing the corrections/missing parts. REFLECT and IMPROVE the script based on your reflection. Find and remove any GAML errors. \*\*THEN OUTPUT THE FULL, COMPLETE AND VALID GAML SCRIPT\*\* and memorise as {key-gamlScript}.

# Conclusion

* Now, write a new, different Markdown document (separate each section using headers). Only show the final, resulting markdown file code from this prompt. First, output "Conclusion" as Markdown 'Heading Level 3 ###'. Then: Write a 300 WORD (if possible) conclusion of the entire conversation history. Provide 3 paragraphs, testifying whether the aim has been achieved, refer to and answer the memorised {key-objectives} and memorised {key-hypotheses} in your conclusion. Also, provide 2 identified limitations of the current work, and propose 2 ideas for future work, based on these limitations. Also mention what the memorised {key-gamlScript} achieves and how it fits into the memorised {key-hypotheses} and memorised {key-objectives}. Memorise this conclusion as {key-conclusion}.